An Introduction of Geometric Programming Using Gate Sizing

Jozeanne Belomo, Gracieli Posser, Guilherme Flach, Ricardo Reis
{ jbelomo,gposser,gaflach,reis}@inf.ufrgs.br

Universidade Federal do Rio Grande do Sul (UFRGS)
Instituto de Informática – PPGC/PGMicro
Av. Bento Gonçalves 9500 Porto Alegre, RS - Brazil

Abstract

Geometric Programming (GP) is a method to solve non-linear problems. In this paper, the gate sizing problem to minimize delay is used as example of formulation using Geometric Programming. Two solvers are used to solve the gate sizing problem, GGPLAB and MOSEK. GGPLAB is a Matlab toolbox to solve Geometric Programs and generalized geometric programs. MOSEK is a software which solves large-scale mathematical optimization problems and it has interface to a group of programming languages. The runtime of the MOSEK is 89.6% lower, on average, than GGPLAB to solve the gate sizing problem considering a set of the ISCAS'85 benchmark circuits.

1. Introduction

Geometric programming (GP) is a technique to solve nonlinear optimization problems. A Geometric Program (GP) in standard form may become a convex program after a logarithmic transformation. The global optimum of a convex problem is achieved more quickly than the result of a non-linear problem.

Since its inception, GP has been closely associated with applications in engineering. Early applications of geometric programming include nonlinear network flow problems, optimal control, optimal location problems, and chemical equilibrium problems [1].

With the development of methods that can solve large-scale GPs extremely efficiently and reliably; at the same time a number of practical problems, particularly in circuit design, have been found to be equivalent to (or well approximated by) GPs. The basic approach in GP modeling is to attempt to express a practical problem, such as an engineering analysis or design problem, in GP format. In the best case, this formulation is exact; when this is not possible, we settle for an approximate formulation. Placement, routing, power control, floor planning, wire sizing, gate sizing are some examples of possible problems that can be formulated as a Geometric Program. In this work, a gate sizing formulation using Geometric Programming is presented as example and the problem is solved using GP solvers [2].

Section II presents some related works that are casted by geometric programming. In section III, Geometric Programming model is shown. The geometric programming solvers are presented in Section IV. Section V shows an application formulated using geometric programming and its runtime results. The chosen application is the gate sizing. Conclusions are drawn in section VI.

2. Related Works

GP can be used to formulate different problems in circuit design. Some methods to transform these problems into GP format are found in [1]. Placement is addressed in [3] to improve delay in integrated circuits where the technique lies a mathematical formulation for simultaneous cell sizing and placement subject to timing and position constraints. [4] employ geometric programming (GP) for the optimization problem of temperature reduction and chip area floorplanning. [5] presents a method for digital circuit optimization based on formulating the problem as a geometric program, it is considered gate and wire designs and some complex formulations as such robust design over corners, multimode design, statistical design, and problems in which threshold and power supply voltage are also variables to be chosen.

The gate-sizing problem using geometric programming has been studied in many works. In [6], the Elmore delay model is reduced to unary geometric programs with the objective of improve the performance of VLSI circuits sizing their gates. The unary GP is solved using a greedy algorithm with runtime linear to the number of components in the circuit. [7] has a convex programming approach to solve the transistor sizing problem. Delay can be controlled by varying the sizes of transistors in a circuit, spending additional chip area. Delay is modeled as posynomials functions of transistor size, creating a geometric programming problem which is transformed into a convex program, guarantying to find the exact solution. In [8], an optimization scheme for gate sizing in the presence of process variations using a posynomial delay model is presented and the resulting optimization

problem is relaxed to be a Geometric Program that is solved using convex optimization tools. And, GP is used in delay model for transistor sizing [9].

3. Geometric Programming Model

The term geometric program was introduced in 1967 [10] and is used to define a type of mathematical optimization problem whose objective must be a posynomial function. There is a difference between the terms geometric programming and geometric optimization. Geometric Programming refers to the family of optimization problems and Geometric optimization refers to optimization problems involving geometry [1].

To understand what is a posynomial function, we explain what is a monomial function. In a monomial function, as shown in equation (1), *c* is the coefficient and it can be any positive number. The *c* coefficient multiplies the variables with exponents and the exponents can be any real numbers.

$$f(x)^{\gamma} = (cx_1^{a1}x_2^{a2}...x_n^{an})^{\gamma} = c^{\gamma}x_1^{\gamma a1}x_2^{\gamma a2}...x_n^{\gamma an} \quad (1)$$

The sum of one or more monomials, that is, the function as

$$f(x) = \sum_{K=1}^{K} c_k x_1^{a1k}x_2^{a2k}...x_n^{ank} \quad (2)$$

where $c_k > 0$, is called a posynomial function (with K terms, in the variables *x1*, ..., *xn*).

Any monomial is also a posynomial. Posynomials are closed under addition, multiplication, and positive scaling. Posynomials can be divided by monomials resulting also a posynomial [1].

Geometric Programming (GP) is an optimization problem of the form

$$\begin{aligned} &\text{minimize } f0(x) \\ &\text{subject to } fi(x) \leq 1,\ i = 1, \ldots, m, \\ &\qquad gi(x) = 1,\ i = 1, \ldots, p, \end{aligned} \quad (3)$$

where *fi* are posynomial functions, g_i are monomials, and *xi* are the optimization variables. We refer to the problem 3 as a geometric program in standard form. In a standard form geometric program, the objective must be posynomial (and it must be minimized); the equality constraints can only have the form of a monomial equal to one, and the inequality constraints can only have the form of a posynomial less than or equal to one [1].

4. Geometric Programming Solvers

GP solvers unambiguously determine feasibility. But they differ in what point (if any) they return when a GP is determined to be infeasible. In any case, it is always possible to set up and solve the problems described above (or others) to find a potentially useful 'nearly feasible' point.

4.1. GGPLAB

GGPLAB is a Matlab-based toolbox for specifying and solving geometric programs (GPs) and generalized geometric programs (GGPs) [11].

This toolbox accepts GP in posynomial and convex form. GGPLAB consists of *gpcvx*, a primal-dual interior-point solver for GP (in convex form) and a wrapper, *gpposy*, which accepts GPs given in posynomial form. A library of objects to support the specification of GPs and GGPs and a variety of examples is available at GGPLAB file. The solver supports sparse problems, but it is not optimized for large scale problems [11].

4.2. MOSEK

MOSEK optimization software is designed to solve large-scale mathematical optimization problems. It includes interfaces to C/C++, Java, MATLAB, .NET(C#/Visual Basic) and Python.

MOSEK is designed to solve linear problems, conic quadratic problems, general convex problems, quadratic problems, convex quadratically constrained problems, geometric problems (posynomial case) and integer problems. To solve these problems, MOSEK uses interior-point optimizer, primal or dual simplex optimizer, conic interior-point optimizer and Mixed-integer optimizer [12].

The solver transforms a GP input in a convex optimization problem and solves it. The output of the convex problem uses the same method to have the GP result, a simple variable transformation.

MOSEK provides faculty, students, or staff of a degree-granting academic institution a free unlimited size license. The free academic license can only be used for research or educational purposes, that is our case [12].

5. Gate Sizing using Geometric Programming

Geometric Programming was applied in a gate sizing tool. Equation (4) shows the formulation of the gate sizing optimization problem to minimize delay, considering that D values are the delays of the circuit paths. X_{min}

and X_{max} are the minimum and maximum size of the gate (gate sizing) or the minimum and maximum allowed transistors width (transistor sizing). The maximum transistor width was defined by the maximum transistor width found in a cell library with a same manufacturing technology. $C_{in}^{\max}$ is the maximum input capacitance acceptable to the circuit, avoiding a high load to its driver and A_{max} is the maximum circuit area [2].

$$\begin{aligned} &\text{minimize} && D = max(D_1 ... D_n) \\ &\text{subject to} && X_{\min} \le X_i \le X_{\max} \\ & && C_{in} \le C_{in}^{max} \qquad A \le A_{max} \end{aligned} \tag{4}$$

The gate sizing tool solves the problem in this way [2]:

1) The logic gates are modeled using the Switch-Level RC Gate Model [5]. In this model, a gate is viewed as a set of RC trees one for each possible input vectors and the gate delay is the maximum delay generated by its compound RC trees.

2) Delay is calculated using the Elmore delay model [13], which produces posynomial functions, enabling the problem solution by Geometric Programming.

3) Area is the sum of the width W_i of each transistor i that make up the circuit, where n is the number of transistors of the circuit keeping the GP form:

$$A_{total} = \sum_{i=1}^{n} W_i \tag{5}$$

4) Switching power is calculated considering all capacitances of the circuit: load capacitance (C_{load}) and input capacitances (C_{in}). *Vdd* is the circuit voltage. **α** is the switching probability, which we consider 20% and f is the clock frequency, which we set to 500MHz for our tests.

$$P = (C_{load} + \sum_{i=1}^{n} Cin_i) * Vdd^2 * \alpha * f \tag{6}$$

5.1. Runtime Results

The gate sizing tool to minimize delay of the circuits is used to size a set of the ISCAS'85 benchmark circuits [2]. Tab. 1 shows a runtime comparison using both GP solvers mentioned in section 4, GGPLAB and Mosek, and the runtime reduction achieved by Mosek. Tab. 1 also shows for each circuit the number of gates, the number of auxiliary variables and the total number of variables used to solve the gate sizing problem. Auxiliary variables are used to translate the generalized geometric problem to the standard form. The total number of variables for the gate sizing is the sum of the number of gates and the number of auxiliary variables.

Mosek solver is able to reduce the runtime to size the gates in 89.6%, on average, producing the same area, timing and power results than GGPLAB solver.

Tab. 1 – Runtime in seconds considering the gate sizing using two different solvers: GGPLAB as a toolbox of Matlab and Mosek

	# Gates	# Auxiliary Variables	# Variables	Runtime (s)		
				GGPLAB	Mosek	R (%)
C432	184	344	528	74.5	8.6	**88.49**
C499	403	755	1158	307.2	14.4	**95.30**
C190			714	127.5	9.2	**92.78**
8	259	455				
C880	232	399	631	75.9	8.1	**89.39**
apex1	1728	3351	5079	3021.5	369.7	**87.76**
apex2	4110	8003	12113	23708	3996.5	**83.14**
apex3	1939	3771	5710	3927.8	455.4	**88.41**
apex5	1942	3663	5605	4816.9	412.3	**91.44**
Avg.	1350	2593	3942	4507.4	659.3	**89.6**

6. Conclusions and Future Works

In this paper, the Geometric Programming (GP) technique was presented. Two solvers that can be used to solve the GP problems were shown too.

Gate sizing was used as example of GP formulation. To solve the gate sizing problem, GGPLAB and MOSEK solvers were used. The two solvers get the same optimization results, but the runtime of MOSEK was

considerably lower. This happen because MOSEK is designed to resolve large scale problems and it is not used as a toolbox of other tool, as GGPLAB that is a Matlab toolbox.

As a future work, we intend to use Geometric Programming to solve the gate sizing problem taking into account other design variables, like rise and fall times to calculate the delay and consider leakage power as a constraint in the problem formulation.

7. Acknowledgment

This work is partially supported by Brazilian National Council for Scientific and Technological Development (CNPq–Brazil) and Coordination for the Improvement of Higher Education Personnel (CAPES).

8. References

[1] S. Boyd, S.-J. Kim, L. Vandenberghe, and A. Hassibi, "A tutorial on geometric programming," *Springer Science+Business Media*, pp. 67– 127, 2007.

[2] G. Posser, G. Flach, G. Wilke, R. Reis. "Gate Sizing Minimizing Delay and Area," in IEEE Computer Society Annual Symposium on VLSI, ISVLSI 2011, Chennai, India, 2011.

[3] W. Chen, C-T. Hseih, M. Pedram. "Simultaneous gate sizing and placement," IEEE Transactions on Computer Aided Design of Integrated Circuits and Systems 19(2):206–214, feb. 2000.

[4] Y-C. Chen, Y. Li, "Temperature-aware floorplanning via geometric programming" Mathematical and Computer Modelling, Vol.51, No.7-8, pp. 927-934, April 2010.

[5] S.Boyd, S.-J. Kim, D. Patil, and M. Horowitz, "Digital circuit optimization via geometric programming," *Operations Research*, vol. 53, no. 6, pp. 899–932, Nov.-Dec. 2005.

[6] C. Chu and D.Wong, "VLSI circuit perfomance optimization by geometric programming", Annals of Operations Research, vol 105, pp. 37-60, 2001.

[7] S. Sapatnekar, V. Rao, P. Vaidya, and S.-M. Kang, "An exact solution to the transistor sizing problem for cmos circuits using convex optimization," *IEEE Transactions on Computer Aided Design of Integrated circuits and Systems*, vol. 12, no. 11, pp. 1621–1634, 1993.

[8] J. Singh, V. Nookala, Z-Q Luo, S. Sapatnekar, "Robust Gate Sizing by Geometric Programming," In: Proceedings of the 42nd IEEE/ACM Design Automation Conference (DAC), pp 315–320, 2005.

[9] M. K. K. Kasamsetty and S. S. Sapatnekar, "A new class of convex functions for delay modeling and their application to the transistor sizing problem," IEEE Trans. on Computer-Aided Design of Integrated Circuits and Systems, vol. 19, no. 7, pp. 779–788, 2000.

[10] R. J. Duffin, E. L. Peterson, and C. Zener, "Geometric programming- theory and application," *John Wiley & Sons*, 1967.

[11] A. Mutapcic, K. Koh, S. Kim and Stephen Boyd, "ggplab version 1.00 A Matlab Toolbox for Geometric Programming", 2006, available in http://stanford.edu/~boyd/ggplab/ggplab.pdf.

[12] E. D. Andersen and K. D. Andersen, "The MOSEK C API manual.Version 6.0 (Revision 122)" Available in <http://docs.mosek.com/6.0/tools.pdf> and the software is available in <http://mosek.com>

[13] W. Elmore, "The transient analysis of damped linear networks with particular regard to wideband amplifiers," *J. Applied Physics*, vol. 19, 1948.